대 한 민 국 특 허 청
KOREAN INDUSTRIAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Industrial Property Office.

출 원 번 호 : 특허출원 1999년 제 43749 호
Application Number

출 원 년 월 일 : 1999년 10월 11일
Date of Application

출 원 인 : 삼성전자 주식회사
Applicant(s)

2000 년 05 월 02 일

특 허 청
COMMISSIONER

【서류명】 특허출원서
【권리구분】 특허
【수신처】 특허청장
【제출일자】 1999.10.11
【국제특허분류】 G06F 9/46
【발명의 명칭】 디지털 시그널 프로세서를 위한 실시간 제어 시스템
【발명의 영문명칭】 Real-time Control System for Digital Signal Processor
【출원인】
【명칭】 삼성전자 주식회사
【출원인코드】 1-1998-104271-3
【대리인】
【성명】 임평섭
【대리인코드】 9-1998-000438-0
【포괄위임등록번호】 1999-007182-1
【발명자】
【성명의 국문표기】 명윤찬
【성명의 영문표기】 MYUNG,Yun Chan
【주민등록번호】 730123-1626111
【우편번호】 135-271
【주소】 서울특별시 강남구 도곡1동 954-1번지 103호
【국적】 KR
【취지】 특허법 제42조의 규정에 의하여 위와 같이 출원합니다. 대리인 임평섭 (인)
【수수료】
【기본출원료】 20 면 29,000 원
【가산출원료】 8 면 8,000 원
【우선권주장료】 0 건 0 원
【심사청구료】 0 항 0 원
【합계】 37,000 원

【요약서】

【요약】

본 발명은, 멀티태스킹 환경에서 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 우선순위 링크를 우선순위 갯수만큼 각각 포함하는 레디 큐(ready queue) 및 웨이팅 큐(waiting queue)를 통해 실시간적으로 태스크들 간의 스위칭과 메시지 전달을 각각 수행하고, 다수의 타이머가 존재하더라도 결정적인 시간 특성을 갖도록 하기 위해 포인터 배열 형태의 두 개의 타이머 휠(timer wheel)을 통해 타이머 제어 블록(timer control block)을 관리하며, 사용자 요구에 따라 고속을 원할 경우엔 내부 메모리를 할당하고 그렇지 않은 경엔 외부 메모리를 할당하고, 내부 메모리가 모두 할당됐으면 상기 외부 메모리를 이용하도록 메모리 관리를 수행하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템에 관한 것이다.

본 발명에 따르면, 비결정론적인 시간 특성을 최대한 제거하여 디지털 시그널 프로세서를 위한 실시간적인 시스템 제어를 수행할 수 있는 이점이 있다.

【대표도】

도 2

【색인어】

DSP, RTOS, 레디 큐, 웨이팅 큐, 타이머 휠

【명세서】

【발명의 명칭】

디지털 시그널 프로세서를 위한 실시간 제어 시스템{Real-time Control System for Digital Signal Processor}

【도면의 간단한 설명】

도 1a 및 도 1b는 각각 종래의 실시간 운영 체계 커널의 레디 큐와 메모리를 설명하기 위한 예시도,

도 2는 본 발명에 따른 디지털 시그널 프로세서를 위한 실시간 제어 시스템의 바람직한 실시예를 나타낸 블록도,

도 3은 본 발명에 따른 레디 큐의 구조의 일례를 나타낸 예시도,

도 4는 본 발명에 따른 웨이팅 큐의 구조의 일례를 나타낸 예시도,

도 5는 본 발명의 메모리 구조의 일례를 나타낸 예시도,

도 6은 본 발명의 타이머 훨 구조의 일례를 예시한 예시도이다.

<도면 주요 부분에 대한 부호의 설명>

10 : 레디 큐	20 : 웨이팅 큐
30 : 내부 메모리	40 : 외부 메모리군
50 : 타이머 훨	60 : 시스템 제어부

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<11> 본 발명은 디지털 시그널 프로세서를 위한 실시간 제어 시스템에 관한 것으로, 더욱 상세하게는 실시간 데이터 처리로 대변되는 디지털 시그널 프로세서(DSP; Digital Signal Processor)의 특징들을 최대한 지원하는 결정론적 실시간 운영 체계 커널을 구현함으로써 좀 더 개념적으로 편리하게 DSP 어플리케이션을 작성하고 원하는 정확한 결과를 얻을 수 있도록 한 DSP용 실시간 제어 시스템에 관한 것이다.

<12> DSP(Digital Signal Processor)의 시장은 갈수록 확대되고 있으며 범용 프로세서(general purpose processor)의 성장 속도를 추월할 정도로 응용 분야가 넓어지고 있다. 현재의 정보 통신 분야의 발달 및 앞으로 디지털 가전 시장의 성장을 고려해 볼 때 무한한 발전 가능성을 내다 볼 수 있는 분야이다. 이와 같은 DSP는 디지털 신호 처리 알고리즘을 구현해야 하는 특성상 소프트웨어가 많은 비중을 차지한다. 또한, 수치 연산 처리가 많이 요구되기 때문에 여러 개의 작업을 동시에 진행해야 하는 산업용 내장 시스템(embedded system)에서는 운용 체계(operating system)의 사용이 필수적으로 대두된다. 신호 처리가 대부분 실시간으로 구현되어야 하기 때문에 기존의 범용 운용 체계들이 가지는 구조는 적합하지 않으며 실시간 특성을 구현하면서 DSP가 가지는 특징들을 지원할 수 있는 DSP용 실시간 커널(real-time kernel)의 등장이 요구되고 있다. 종래의 실시간 커널들의 동작을 큐(queue)와 관련된 부분과 메모리 관리와 연관된 부분으로 나누어 살

펴보기로 한다.

<13> 도 1a 및 도 1b는 각각 종래의 실시간 운영 체계 커널의 레디 큐와 메모리를 설명하기 위한 예시도이다.

<14> 먼저, 종래의 실시간 운영 체계 커널의 레디 큐(ready queue)는 도 1a와 같이 이중 링크드 리스트 구조를 갖는다. 실시간적 처리를 요구하지 않는 운영 체계에서는 멀티프로그래밍(multiprogramming)을 수행함에 있어서, 통상, 레디 큐는 라운드로빈(round robin) 방식으로 태스크를 지정하지만 DSP와 같이 실시간 운영 체계가 필요한 시스템에서는 이중 링크드 리스트 구조의 이점인 순방향 링크(flink)와 역방향 링크(blink)의 데이터 구조의 특성을 충분히 살리면서 태스크 제어 블록을 빠르게 검사하여 시간적으로 우선순위를 갖는 태스크를 먼저 수행할 수 있도록 배려한다. 웨이팅 큐(waiting queue) 역시 이와 유사하게 이중 링크드 리스트(double linked list) 구조를 갖는다.

<15> 이하, 스케줄링(scheduling)을 위해 최우선순위의 태스크(task, 작업)를 찾는 과정을 설명하면 다음과 같다.

<16> 우선, 레디 큐가 가리키는 태스크 제어 블록(TCB; Task Control Block)을 검토하여 최우선순위와 일치하는지 검토한 후, 일치하지 않으면 링크를 따라 다음 태스크 제어 블록(TCB)을 검토하며 더 이상 순방향 링크(flink)가 연결되지 않거나 최우선순위와 일치하는 태스크 제어 블록이 발견될 때까지 이상의 과정을 반복한다. 여기서, 태스크 제어 블록이란 태스크에 대응하여 두어지고 그 태스크에 관련된 정보나 자원의 상태를 나타내는 정보를 포함하여 태스크의 제어에 쓰이는 블록을 말한다.

<17> 도 1b를 참조하여 설명하면, 기존의 대부분의 내장 시스템의 메모리 관리는 가상

메모리(virtual memory)를 사용하지 않고 실제 메모리(physical memory)와 논리 메모리(logical memory)를 일대일로 대응시켜 사용한다. 여러 가지 형태의 메모리가 혼재되어 있더라도 특별히 구분하여 사용하는 경우는 ROM(Read Only Memory)에 국한되고 나머지는 사용자의 요구에 따라 구분 없이 할당되는 것이 보통이다. 개발 과정일 경우, 통상, ROM에는 모니터 프로그램(monitor program)이나 다운로딩(downloading)과 디버깅(debugging)을 위한 기본적인 운영 체계가 저장되고 RAM(Random Access Memory)에는 실제 동작 코드가 존재하게 된다. 가령, 사용자 요구에 따라 고속을 원할 경우엔 상기 내부 메모리를 할당하고 그렇지 않은 경엔 외부 메모리를 할당한다든지 하는 별도의 고려를 하지 않기 때문에 주어진 메모리 자원을 충분히 활용하지 못하는 단점이 있다.

<18> 앞서 설명한 내용을 통해 알 수 있듯이, 종래의 경우, 커널 내부에서 사용되는 큐의 구조는 비결정론적(non-deterministic)임을 알 수 있다. 즉, 연결된 태스크의 갯수에 따라 큐잉(queuing)을 수행하는 시간이 매우 가변적이라는 것이다. 다시 말해서, 연결된 태스크의 갯수가 많을수록 이에 비례하여 검색에도 많은 시간이 소요된다. 해싱(hashing) 등을 사용하기도 하지만, 예컨대, 16비트 DSP에서는 최대 16개의 정도의 태스크가 존재한다고 가정하므로 이는 오히려 연산의 무리를 가져온다. 이것은 태스크들 간의 스위칭에 사용되는 레디 큐뿐만 아니라 태스크들 간 메시지 전달에 사용되는 웨이팅 큐에서도 동일한 문제들이 발생한다. 디지털 신호 처리에서는 처리 시간이 매우 중요한 응용이 많고 심지어는 데드라인 스캐줄링(deadline scheduling)이 요구되기도 하므로 예측 가능한 운영 체계 모델이 필요한 데, 비결정론적 구조를 갖는 종래의 실시간 커널 구조로는 이와 같은 요구에 부합할 수 없는 문제가 있었다.

<19> 또한, 메모리 구조 측면에서 볼 때, DSP는 다중 메모리 구조를 사용한다. 이는 성

능 향상을 위한 것으로, 대부분의 DSP가 채택하고 있는 하버드 구조(Harvard Architecture)로 인해 가능하게 되는 데, 종래의 메모리 관리로는 많은 연산이 요구되는 부분에서 프로세서 내부 메모리를 고속으로 할당할 수가 없음에 따라 결국, 사용자가 운영 체계와는 별도로 메모리 관리를 해야하는 부담이 생기는 문제가 있었다.

【발명이 이루고자 하는 기술적 과제】

<20> 따라서, 본 발명은 이와 같은 문제를 해결하기 위해 안출된 것으로, 정확한 실시간 특성들을 구현하고자 커널이 가지는 비결정론적인 시간 특성을 최대한 제거하기 위해 레디 큐(ready queue)와 웨이팅 큐(waiting queue) 등의 구조를 최적화하고 DSP 프로세서의 다중 어드레스 공간, 모듈러 어드레싱(modular addressing), 서큘러 어드레싱(circular addressing) 등을 지원토록 함으로써 산업용 내장 시스템에서 요구되는 조건들과 DSP에서 요구되는 조건들을 동시에 만족시킬 수 있는 DSP용 실시간 제어 시스템을 제공함에 그 목적이 있다.

【발명의 구성 및 작용】

<21> 이와 같은 목적을 달성하기 위해 본 발명에 따른 디지털 시그널 프로세서를 위한 실시간 제어 시스템은, 멀티태스킹 환경에서 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 우선순위 링크를 우선순위 갯수만큼 각각 포함하는 레디 큐(ready queue) 및 웨이팅 큐(waiting queue)를 통해

실시간적으로 태스크들 간의 스위칭과 메시지 전달을 각각 수행하고, 다수의 타이머가 존재하더라도 결정적인 시간 특성을 갖도록 하기 위해 포인터 배열 형태의 두 개의 타이머 휠(timer wheel)을 통해 타이머 제어 블록(timer control block)을 관리하며, 사용자 요구에 따라 고속을 원할 경우엔 내부 메모리를 할당하고 그렇지 않은 경엔 외부 메모리를 할당하고, 내부 메모리가 모두 할당됐으면 상기 외부 메모리를 이용하도록 메모리 관리를 수행함으로써 비결정론적인 시간 특성을 최대한 제거하여 실시간적인 시스템 제어를 수행할 수 있는 것이 특징이다.

<22> 이하, 본 발명에 따른 디지털 시그널 프로세서를 위한 실시간 제어 시스템의 바람직한 실시예를 첨부한 도면을 참조하여 설명하면 다음과 같다.

<23> 도 2는 본 발명에 따른 디지털 시그널 프로세서를 위한 실시간 제어 시스템의 바람직한 실시예를 나타낸 블록도이다.

<24> 본 발명의 바람직한 실시예는 도 2에 나타낸 바와 같이, 멀티태스킹 환경에서 태스크들 간의 스위칭을 위해, 시작 태스크 블록과 마지막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list pointer)와 라스트 포인터(last pointer)를 포함하는 레디 큐 링크(ready queue link)를 구비함으로써 검색 과정 없이 태스크 제어 블록(task control block)을 추가토록 하고, 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 제 1 우선순위 링크(1st priority link)를 우선순위 갯수만큼 구비한 레디 큐(ready queue, 10)와;

<25> 멀티태스킹 환경에서 태스크들 간의 메시지 전달을 위해, 시작 태스크 블록과 마지막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list pointer)와 라스트 포인터(last pointer)를 포함하는 웨이팅 큐 링크(waiting queue link)를 구비함으로써 검색 과정 없이 태스크 제어 블록(task control block)을 추가토록 하고, 동일한 우선순위를 갖는 태스크 제어 블록들 중에서 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 제 2 우선순위 링크(2nd priority link)를 우선순위 갯수만큼 구비한 웨이팅 큐(waiting queue, 20)와;

<26> 자신의 시작 어드레스, 자신의 종료 어드레스, 메모리 크기 정보와 메모리 맵 정보 및 다음에 연결될 외부 메모리를 가리키는 넥스트 정보로 구성된 구조체 구조를 통해 관리되는 내부 메모리(30)와;

<27> 자신의 시작 어드레스, 자신의 종료 어드레스, 메모리 크기 정보와 메모리 맵 정보 및 다음에 연결될 다른 외부 메모리를 가리키는 넥스트 정보로 구성된 구조체 구조를 통해 관리되는 외부 메모리을 적어도 하나 이상 구비한 외부 메모리군(40)과;

<28> 다수의 타이머가 존재하더라도 결정적인 시간 특성을 갖도록 하기 위해 포인터 배열 형태로 타이머 제어 블록(timer control block)을 관리하는 두 개의 타이머 휠(timer wheel, 50); 및

<29> 태스크들 간의 스위칭과 메시지 전달을 위해 각각 레디 큐(10)와 웨이팅 큐(20)를 제어하여 스케줄링을 수행하고, 사용자 요구에 따라 고속을 원할 경우엔 내부 메모리(30)를 할당하고 그렇지 않은 경엔 외부 메모리군(40)을 할당하고, 내부 메모리(30)가 모두 할당됐으면 상기 외부 메모리군(40)을 이용하도록 메모리 관리를 수행하며, 태스크

에 설정된 시간값에 따라 기준 시간 이하일 경우에는 첫 번째 타이머 휠의 해당 슬롯에 삽입하고 상기 기준 시간 보다 크지만 상기 기준 시간의 2배 이하일 경우에는 두 번째 타이머 휠의 해당 슬롯에 삽입하도록 제어함으로써 비결정론적인 시간 특성을 최대한 제거하여 실시간적인 시스템 제어를 수행하는 시스템 제어부(60)를 포함하도록 구성된다.

<30> 이와 같이 구성된 본 발명에 따른 디지털 시그널 프로세서를 위한 실시간 제어 시스템의 바람직한 실시예의 작용을 첨부한 도면을 참조하여 상세하게 설명하기로 한다.

<31> 도 3은 본 발명에 따른 레디 큐(10)의 구조의 일례를 나타낸 예시도이다.

<32> 우선, 레디 큐(10)는 도 3에 도시한 바와 같이, 레디 큐 링크(readyQ_link)와 제 1 우선순위 링크(t_priority_link)를 구비한다. 레디 큐(10)의 레디 큐 링크(readyQ_link)는 시작 태스크 블록과 마지막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list)와 라스트 포인터(last)를 포함으로써 검색 과정 없이 태스크 제어 블록(TCB)을 추가하는 것이 가능하다. 또한, 레디 큐(10)의 제 1 우선순위 링크(t_priority_link)는 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록을 가리키는 포인터(list)와 종료 태스크 제어 블록을 가리키는 포인터(last)로 구성됨에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있다.

<33> 도 3을 참조할 때, 멀티태스크에 대한 스케줄링이 필요할 시, 태스크들 간의 스위칭을 위해 시스템 제어부(60)는 태스크 제어 블록이 가지는 우선순위에 따라 태스크 제어 블록을 상기 제 1 우선순위 링크(t_priority_link)에 삽입하거나 또는 상기 제 1 우선순위 링크(t_priority_link)로부터 삭제되도록 하면서 상기 레디 큐 링크

(readyQ_link)의 구조가 유지될 수 있도록 하기 위해 상기 제 1 우선순위 링크(t_priority_link)에 태스크 제어 블록이 삽입된 경우엔 삽입된 태스크 제어 블록을 상기 레디 큐 링크(readyQ_link)에 연결시키고 태스크 제어 블록이 삭제된 경우엔 남아있는 태스크 제어 블록을 상기 레디 큐 링크(readyQ_link)에 연결하도록 상기 레디 큐(10)를 제어하여 스케줄링을 수행한다.

<34> 다음으로, 도 4는 본 발명에 따른 웨이팅 큐(20)의 구조의 일례를 나타낸 예시도이다.

<35> 웨이팅 큐(20)는 도 4에 도시한 바와 같이, 웨이팅 큐 링크(readyQ_link)와 제 2 우선순위 링크(t_priority_link)를 구비한다. 웨이팅 큐(20)의 웨이팅 큐 링크(waitingQ_link)는 시작 태스크 블록과 마지막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list)와 라스트 포인터(last)를 포함으로써 검색 과정 없이 태스크 제어 블록(TCB)을 추가하는 것이 가능하다. 또한, 웨이팅 큐(20)의 제 2 우선순위 링크(t_priority_link)는 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록을 가리키는 포인터(list)와 종료 태스크 제어 블록을 가리키는 포인터(last)로 구성됨에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있다.

<36> 도 4를 참조할 때, 멀티태스크에 대한 스케줄링이 필요할 시, 태스크들 간의 메시지 전달을 위해 시스템 제어부(60)는 태스크 제어 블록이 가지는 우선순위에 따라 태스크 제어 블록을 상기 제 2 우선순위 링크(t_priority_link)에 삽입하거나 또는 상기 제 2 우선순위 링크(t_priority_link)로부터 삭제되도록 하면서 상기 웨이팅 큐 링크(readyQ_link)의 구조가 유지될 수 있도록 하기 위해 상기 제 2 우선순위 링크(t_priority_link)에 태스크 제어 블록이 삽입된 경우엔 삽입된 태스크 제어 블록을 상

기 웨이팅 큐 링크(readyQ_link)에 연결시키고 태스크 제어 블록이 삭제된 경우엔 남아 있는 태스크 제어 블록을 상기 웨이팅 큐 링크(readyQ_link)에 연결하도록 상기 웨이팅 큐(20)를 제어하여 스케줄링을 수행한다.

<37> 이때, 시스템 제어부(60)는 태스크들 간의 스위칭을 위한 우선순위 기반의 태스크 검색일 경우에는 도 3의 제 1 우선순위 링크(t_priority_link)를 통해 태스크 제어 블록을 검색하고 태스크들 간의 스위칭을 위한 선입선출(FIFO) 기반의 태스크 검색일 경우엔 상기 레디 큐 링크(readyQ_link)를 통해 최선두의 태스크 제어 블록을 가져오도록 제어한다. 한편, 태스크들 간의 메시지 전달을 위한 우선순위 기반의 태스크 검색일 경우에는 도 4의 제 2 우선순위 링크(t_priority_link)를 통해 태스크 제어 블록을 검색하고 태스크들 간의 메시지 전달을 위한 선입선출(FIFO) 기반의 태스크 검색일 경우엔 상기 웨이팅 큐 링크(waitingQ_link)를 통해 최선두의 태스크 제어 블록을 가져오도록 제어한다.

<38> 도 5는 본 발명의 메모리 구조의 일례를 나타낸 예시도이다.

<39> 내부 메모리(30, IRAM)는 도 5에 예시한 바와 같이, 자신의 시작 어드레스(start), 자신의 종료 어드레스(end), 메모리 크기 정보(size)와 메모리 맵 정보(map) 및 다음에 연결될 외부 메모리(ERAMLO)를 가리키는 넥스트 정보(next)로 구성된 구조체 구조를 통해 관리되고, 외부 메모리군(40)은 자신의 시작 어드레스(start), 자신의 종료 어드레스(end), 메모리 크기 정보(size)와 메모리 맵 정보(map) 및 다음에 연결될 다른 외부 메모리(ERAMHI)를 가리키는 넥스트 정보(map)로 구성된 구조체 구조를 통해 관리되는 외부 메모리(ERAMLO, ERAMHI)를 적어도 하나 이상 구비하도록 구성된다.

<40> 메모리 관리를 위해 시스템 제어부(60)는 메모리 할당과 반환을 시스템 콜 형태로

제공하고 사용자 요구에 따라 고속을 원할 경우엔 상기 내부 메모리(30)를 할당하고 그렇지 않은 경엔 상기 외부 메모리(ERAMLO, ERAMHI) 중의 어느 하나의 외부 메모리를 할당하고, 상기 내부 메모리가 모두 할당됐으면 상기 외부 메모리군(40)을 이용하고, 메모리 할당과 반환 과정에서 생기는 단편화(fragmentation)를 최소화하기 위해 일정 크기의 페이지 단위로 메모리 할당과 반환을 수행하고, 할당된 메모리 공간을 관리하기 위해 비트맵을 사용하여 메모리의 할당 및 반환 여부를 확인한다. 한편, 문맥 교환(context switching) 시, 메모리 공간에는 DSP의 특징인 모듈러 어드레싱(modular addressing)과 서큘러 어드레싱(circular addressing)을 지원하기 위해 어드레싱 모드(addressing mode)와 포인터(pointer)가 저장됨은 주지의 사실이다.

<41> 도 6은 본 발명의 타이머 휠 구조의 일례를 예시한 예시도이다.

<42> 본 발명의 바람직한 실시예는 다수의 타이머가 존재하더라도 결정적인 시간 특성을 갖도록 하기 위해 포인터 배열 형태로 타이머 제어 블록(timer control block)을 관리하는 두 개의 타이머 휠(timer wheel, 50)을 구비한다.

<43> 타이머 제어를 위해 시스템 제어부(60)는 태스크에 설정된 시간값에 따라 기준 시간(예컨대, 640㎲) 이하일 경우에는 첫 번째 타이머 휠(도 5에서 1 번째 행)의 해당 슬롯에 삽입하고 상기 기준 시간(640㎲) 보다 크지만 상기 기준 시간의 2배(기준 시간이 640㎲일 때, 1.28㎳) 이하일 경우에는 두 번째 타이머 휠(도 5에서 2 번째 행)의 해당 슬롯에 삽입하며 상기 기준 시간의 2배(1.28㎳)를 초과할 경우에는 에러를 발생시키도록 제어한다.

<44> 기존의 운영 체계에서 스케줄링이 필요할 때, 레디 큐에서 원하는 우선순위의 태스크를 꺼내오거나 태스크들 간 메시지 전달 시 웨이팅 큐에서 원하는 대상 태스크를 꺼내

오는 데에는 예측 불가능한 시간이 걸렸으나 본 발명의 알고리즘으로 커널을 구현함으로써 커널의 동작이 투명해지고 예측 가능하게 된다. 따라서, 신호 처리용 응용 프로그램 개발에 있어 보다 정확한 설계 및 동작이 보장된다. 여기에 부가적으로 원하는 우선순위 태스크에 접근이 한 번에 이루어지기 때문에 커널의 수행 시간을 단축시키는 효과가 더해져 응용 프로그램에 보다 많은 자원을 할당할 수 있는 이점이 있다.

<45> 타이머 역시 기존의 링크드 리스트 방식이 아닌 타이머 휠 방식을 사용함으로써 기존의 방식에서 타이머 만료(timer expiration)가 발생했을 때 해당 태스크를 찾는 데 걸리는 시간을 예측하기 어려웠으나 본 발명은 이러한 단점을 탈피해 이 시간을 예측할 수 있는 이점이 있다.

<46> FIR(Finite Impulse Response) 필터 등을 구현할 때 반복되는 연산 속도를 단축시키기 위해 자주 쓰이는 계수를 프로세서 내부 메모리에 저장해야 다중 메모리 구조와 다중 버스 구조의 도움으로 수행 속도가 향상되지만 종래의 운영 체계에서는 이와 같은 DSP의 특징을 고려하지 않았기 때문에 성능 향상을 기대하기가 어려운 반면에 본 발명에서는 사용자가 선택할 수 있도록 함으로써 최대 성능을 낼 수 있도록 설계의 자유도를 제공하고 있다.

<47> 이와 같은 특징을 바탕으로 결정론적 실시간 운영 체계를 구현할 수 있으며 기존의 운영 체계에서 기대하기 어려운 동작의 투명성이 보장되어 예측 가능하면서도 전체 선능을 향상시키는 DSP 시스템의 설계가 가능하다.

<48> 본원에서 사용되는 용어(terminology)들은 본 발명에서의 기능을 고려하여 정의 내려진 용어들로써 이는 당분야에 종사하는 기술자의 의도 또는 관례 등에 따라 달라질 수 있으므로 그 정의는 본원의 전반에 걸친 내용을 토대로 내려져야 할 것이다.

<49> 또한, 본원에서는 본 발명의 바람직한 실시예를 통해 본 발명을 설명했으므로 본 발명의 기술적인 난이도 측면을 고려할 때, 당분야에 통상적인 기술을 가진 사람이면 용이하게 본 발명에 대한 또 다른 실시예와 다른 변형을 가할 수 있으므로, 상술한 설명에서 사상을 인용한 실시예와 변형은 모두 본 발명의 청구 범위에 모두 귀속됨은 명백하다.

【발명의 효과】

<50> 이상에서 상세하게 설명한 바와 같이, 멀티태스킹 환경에서 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 우선순위 링크를 우선순위 갯수만큼 각각 포함하는 레디 큐(ready queue) 및 웨이팅 큐(waiting queue)를 통해 실시간적으로 태스크들 간의 스위칭과 메시지 전달을 각각 수행하고, 다수의 타이머가 존재하더라도 결정적인 시간 특성을 갖도록 하기 위해 포인터 배열 형태의 두 개의 타이머 휠(timer wheel)을 통해 타이머 제어 블록(timer control block)을 관리하며, 사용자 요구에 따라 고속을 원할 경우엔 내부 메모리를 할당하고 그렇지 않은 경엔 외부 메모리를 할당하고, 내부 메모리가 모두 할당됐으면 상기 외부 메모리를 이용하도록 메모리 관리를 수행하는 본 발명에 따르면, 비결정론적인 시간 특성을 최대한 제거하여 디지털 시그널 프로세서를 위한 실시간적인 시스템 제어를 수행할 수 있는 이점이 있다.

【특허청구범위】

【청구항 1】

멀티태스킹 환경에서 태스크들 간의 스위칭을 위해, 시작 태스크 블록과 마지막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list pointer)와 라스트 포인터(last pointer)를 포함하는 레디 큐 링크(ready queue link)를 구비함으로써 검색 과정 없이 태스크 제어 블록(task control block)을 추가토록 하고, 동일한 우선순위를 갖는 태스크 제어 블록들의 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 제 1 우선순위 링크(1st priority link)를 우선순위 갯수만큼 구비한 레디 큐(ready queue); 및

태스크 제어 블록이 가지는 우선순위에 따라 태스크 제어 블록을 상기 제 1 우선순위 링크에 삽입하거나 또는 상기 제 1 우선순위 링크로부터 삭제되도록 하면서 상기 레디 큐 링크의 구조가 유지될 수 있도록 하기 위해 상기 제 1 우선순위 링크에 태스크 제어 블록이 삽입된 경우엔 삽입된 태스크 제어 블록을 상기 레디 큐 링크에 연결시키고 태스크 제어 블록이 삭제된 경우엔 남아있는 태스크 제어 블록을 상기 레디 큐 링크에 연결하도록 상기 레디 큐를 제어하여 스케줄링을 수행하는 시스템 제어부를 포함하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 2】

제 1 항에 있어서,

멀티태스킹 환경에서 태스크들 간의 메시지 전달을 위해, 시작 태스크 블록과 마

지막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list pointer)와 라스트 포인터(last pointer)를 포함하는 웨이팅 큐 링크(waiting queue link)를 구비함으로써 검색 과정 없이 태스크 제어 블록(task control block)을 추가토록 하고, 동일한 우선순위를 갖는 태스크 제어 블록들 중에서 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 제 2 우선순위 링크(2nd priority link)를 우선순위 갯수만큼 구비한 웨이팅 큐(waiting queue)를 더 포함하고,

상기 시스템 제어부는, 태스크 제어 블록이 가지는 우선순위에 따라 태스크 제어 블록을 상기 제 2 우선순위 링크에 삽입하거나 또는 상기 제 2 우선순위 링크로부터 삭제되도록 하면서 상기 웨이팅 큐 링크의 구조가 유지될 수 있도록 하기 위해 상기 제 2 우선순위 링크에 태스크 제어 블록이 삽입된 경우엔 삽입된 태스크 제어 블록을 상기 웨이팅 큐 링크에 연결시키고 태스크 제어 블록이 삭제된 경우엔 남아있는 태스크 제어 블록을 상기 웨이팅 큐 링크에 연결하도록 상기 웨이팅 큐를 제어하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 3】

제 2 항에 있어서, 상기 시스템 제어부는,

태스크들 간의 스위칭과 메시지 전달을 위한 우선순위 기반의 태스크 검색일

경우에는 각각 상기 제 1 우선순위 링크 및 상기 제 2 우선순위 링크를 통해 태스크 제어 블록을 검색하고, 태스크들 간의 스위칭과 메시지 전달을 위한 선입선출(FIFO) 기반의 태스크 검색일 경우엔 각각 상기 레디 큐 링크 및 상기 웨이팅 큐 링크를 통해 최선두의 태스크 제어 블록을 가져오도록 제어하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 4】

제 1 항에 있어서,

다수의 타이머가 존재하더라도 결정적인 시간 특성을 갖도록 하기 위해 포인터 배열 형태로 타이머 제어 블록(timer control block)을 관리하는 두 개의 타이머 휠(timer wheel)을 더 포함하고,

상기 시스템 제어부는 태스크에 설정된 시간값에 따라 기준 시간 이하일 경우에는 첫 번째 타이머 휠의 해당 슬롯에 삽입하고 상기 기준 시간 보다 크지만 상기 기준 시간의 2배 이하일 경우에는 두 번째 타이머 휠의 해당 슬롯에 삽입하며 상기 기준 시간의 2배를 초과할 경우에는 에러를 발생시키도록 제어하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 5】

제 1 항에 있어서,

자신의 시작 어드레스, 자신의 종료 어드레스, 메모리 크기 정보와 메모리 맵 정

보 및 다음에 연결될 외부 메모리를 가리키는 넥스트 정보로 구성된 구조체 구조를 통해 관리되는 내부 메모리; 및

자신의 시작 어드레스, 자신의 종료 어드레스, 메모리 크기 정보와 메모리 맵 정보 및 다음에 연결될 다른 외부 메모리를 가리키는 넥스트 정보로 구성된 구조체 구조를 통해 관리되는 외부 메모리을 적어도 하나 이상 구비한 외부 메모리군을 더 포함하고,

상기 시스템 제어부는 메모리 할당과 반환을 시스템 콜 형태로 제공하고 사용자 요구에 따라 고속을 원할 경우엔 상기 내부 메모리를 할당하고 그렇지 않은 경엔 상기 외부 메모리 중의 어느 하나의 외부 메모리를 할당하고, 상기 내부 메모리가 모두 할당됐으면 상기 외부 메모리군을 이용하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 6】

제 5 항에 있어서, 상기 시스템 제어부는

메모리 할당과 반환 과정에서 생기는 단편화(fragmentation)를 최소화하기 위해 일정 크기의 페이지 단위로 메모리 할당과 반환을 수행하고, 할당된 메모리 공간을 관리하기 위해 비트맵을 사용하여 메모리의 할당 및 반환 여부를 확인하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 7】

멀티태스킹 환경에서 태스크들 간의 메시지 전달을 위해, 시작 태스크 블록과 마지

막 태스크 제어 블록을 각각 가리키는 리스트 포인트(list pointer)와 라스트 포인터(last pointer)를 포함하는 웨이팅 큐 링크(waiting queue link)를 구비함으로써 검색 과정 없이 태스크 제어 블록(task control block)을 추가토록 하고, 동일한 우선순위를 갖는 태스크 제어 블록들 중에서 시작 태스크 제어 블록과 종료 태스크 제어 블록을 가리킴에 따라 원하는 해당 우선순위의 태스크 제어 블록을 고속으로 인출할 수 있도록 하는 제 2 우선순위 링크(2nd priority link)를 우선순위 갯수만큼 구비한 웨이팅 큐(waiting queue); 및

태스크 제어 블록이 가지는 우선순위에 따라 태스크 제어 블록을 상기 제 2 우선순위 링크에 삽입하거나 또는 상기 제 2 우선순위 링크로부터 삭제되도록 하면서 상기 웨이팅 큐 링크의 구조가 유지될 수 있도록 하기 위해 상기 제 2 우선순위 링크에 태스크 제어 블록이 삽입된 경우엔 삽입된 태스크 제어 블록을 상기 웨이팅 큐 링크에 연결시키고 태스크 제어 블록이 삭제된 경우엔 남아있는 태스크 제어 블록을 상기 웨이팅 큐 링크에 연결하도록 상기 웨이팅 큐를 제어하여 스케줄링을 수행하는 시스템 제어부를 포함하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【청구항 8】

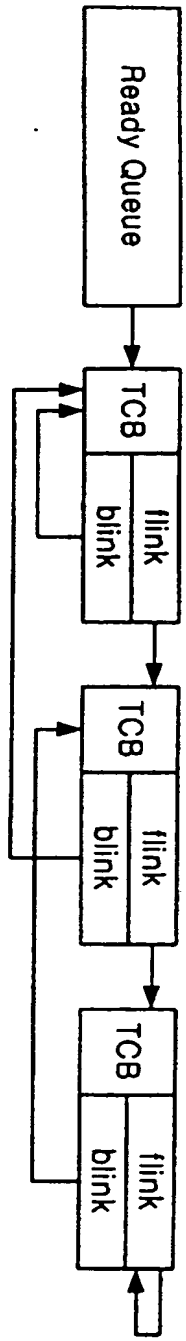
제 7 항에 있어서, 상기 시스템 제어부는,

태스크들 간의 메시지 전달을 위한 우선순위 기반의 태스크 검색일 경우에는 상기 제 2 우선순위 링크를 통해 태스크 제어 블록을 검색하고 태스크들 간의 메시지 전달을 위한 선입선출(FIFO) 기반의 태스크 검색일 경우엔 상기 웨이팅 큐 링크를 통해 최선두

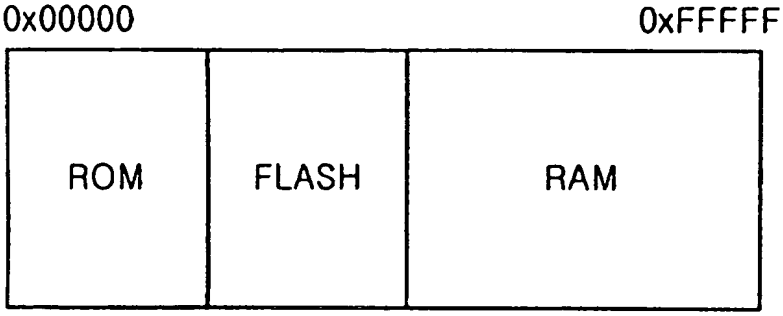
의 태스크 제어 블록을 가져오도록 제어하는 것을 특징으로 하는 디지털 시그널 프로세서를 위한 실시간 제어 시스템.

【도면】

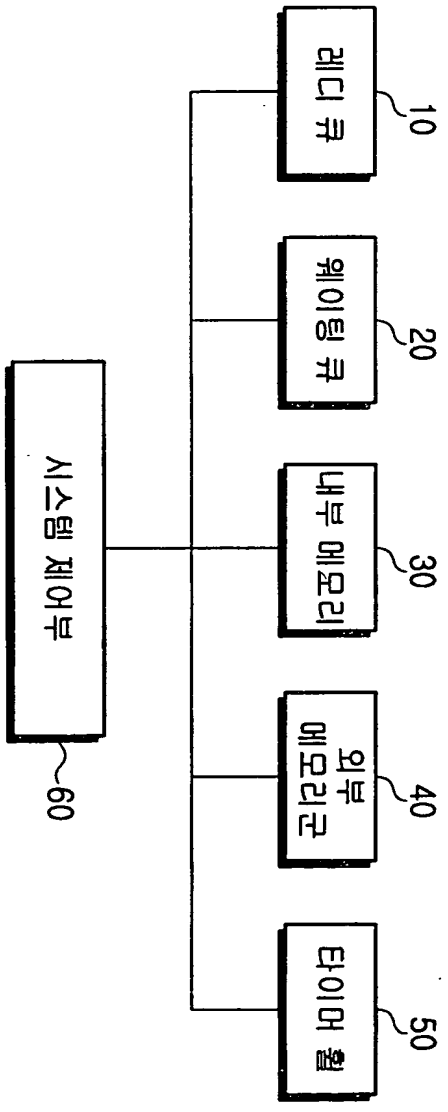
【도 1a】



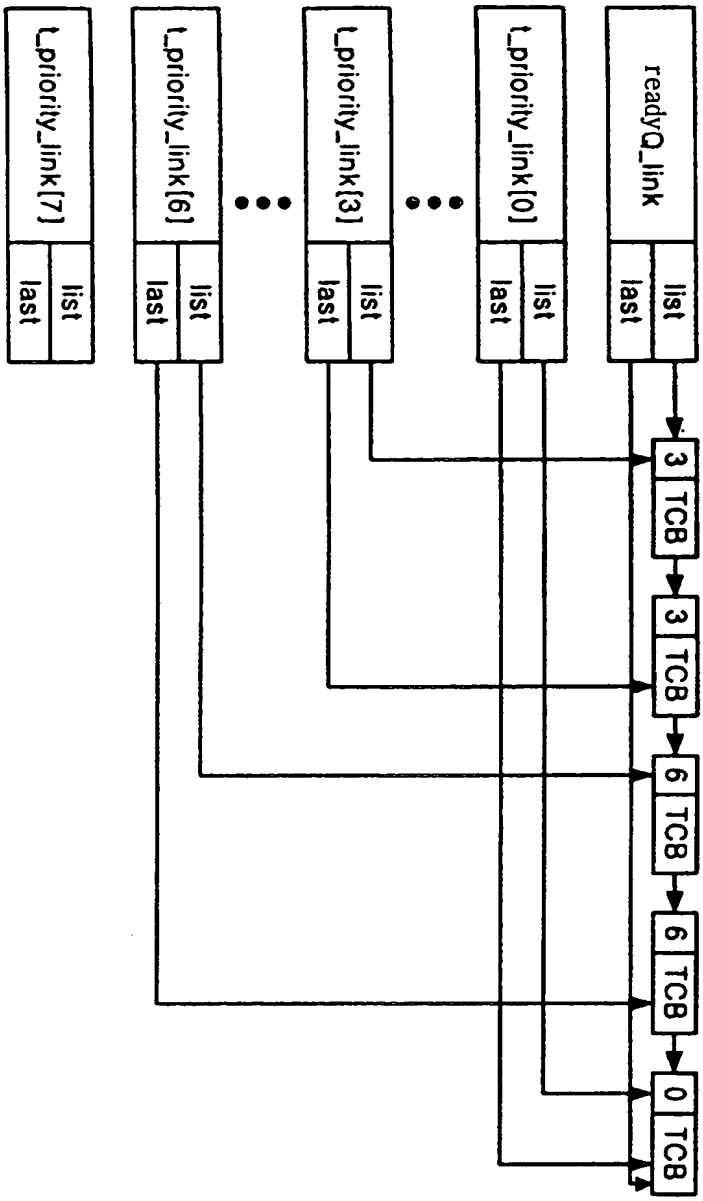
【도 1b】


0x00000
0xFFFFF
ROM
FLASH
RAM

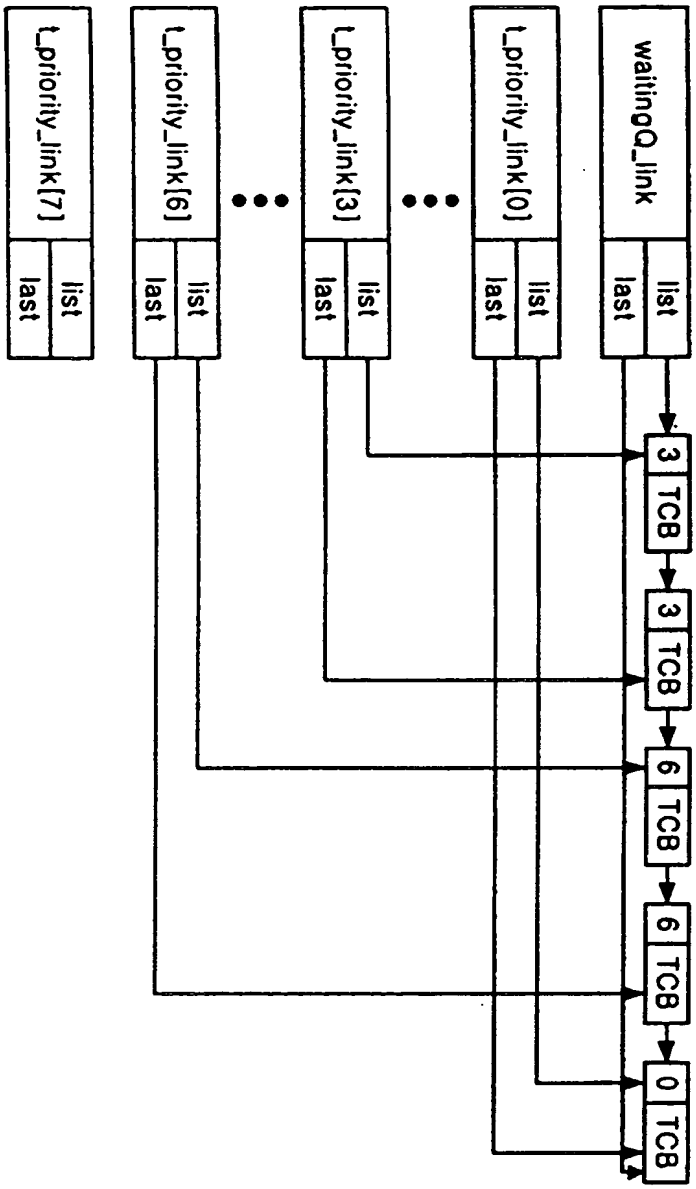
【도 2】


레디 큐
10
웨이팅 큐
20
내부 메모리
30
외부 메모리 관리군
40
타이머 콜
50
시스템 제어부
60

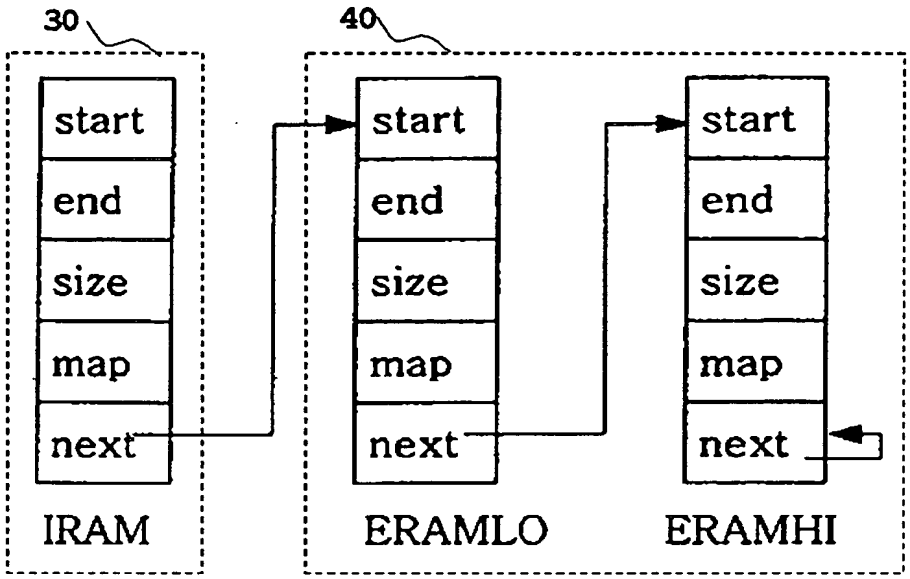
【도 3】


L_priority_link[7]
last
list
L_priority_link[6]
last
list
L_priority_link[3]
last
list
L_priority_link[0]
last
list
readyQ_link
last
list
3
TCB
3
TCB
6
TCB
6
TCB
0
TCB

【도 4】


L_priority_link[7]
last
list
L_priority_link[6]
last
list
L_priority_link[3]
last
list
L_priority_link[0]
last
list
waitingQ_link
last
list
3
TCB
3
TCB
6
TCB
6
TCB
0
TCB

【도 5】



【도 6】

	[0]	[1]		[63]
[0]	struct TMCBHeader	struct TMCBHeader		struct TMCBHeader
[1]	struct TMCBHeader	struct TMCBHeader		struct TMCBHeader